COMPLETE LOGIC PROGRAMS WITH DOMAIN-CLOSURE AXIOM

PAOLO MANCARELLA, SIMONE MARTINI, AND DINO PEDRESCHI

▷ Complete logic programs augmented with the domain-closure axiom are proposed as the reference theory for logic programming with negation as failure. An inference rule corresponding to "proof by case analysis" is proved correct within this framework. As a major consequence, the completeness results for SLD resolution and negation as failure still hold. An interesting outcome is that some novel operational properties of SLD-resolution can be proved. ◁

INTRODUCTION

The *negation-as-failure* rule was introduced to deal with negative information in logic programming [1]. Since a Horn definite logic program has no negative logical consequence, it was necessary to devise a suitable reference theory in order to assign negation-as-failure a declarative, model-theoretic meaning which justifies its use. Thus the notion of *completion* of a logic program, or *complete logic program*, was introduced, which consists of a simple transformation of a logic program P. Roughly speaking, it is a pair (P^*, U^*) where P^* is the theory obtained essentially by replacing the clauses defining each predicate p in P with a single axiom, in which the *if* connectives appearing in the original clauses are replaced by a single *iff* connective. Since this task requires the use of an equality predicate $=$, not appearing in P, P^* is equipped with the theory U^* defining the equality theory induced by *unification*. A complete logic program shares the same positive logical consequences of the corresponding definite program, but it is richer as far as negative information is concerned. Soundness [1] and completeness [2, 14] of negation as failure with respect to (P^*, U^*) have been proved. More precisely, these

Address correspondence to Dr. D. Pedreschi, Dipartimento di Informatica, Università di Pisa, Corso Italia, I-56100 Pisa, Italy.
Received 24 June 1987; accepted 15 October 1987.

THE JOURNAL OF LOGIC PROGRAMMING

52 Vanderbilt Ave., New York, NY 10017 0743-1066/88/$3.50

results can be summarized as follows: given a possibly open literal $p(\underline{t})$, $\forall \neg p(\underline{t})$ is a logical consequence of (P^*, U^*) iff the proof of $p(\underline{t})$ finitely fails under fair SLD resolution ($\underline{t}$ stands for a tuple of terms). In [3] the classical results concerning complete programs are made parametric with respect to the equality theory E^*, i.e., they are extended to generalized logic-programming systems where standard unification, corresponding to syntactic equality, is replaced by some kind of generalized unification, corresponding to some suitable, problem-oriented equality theory E^*.

Relying on the results and proof techniques in [3], the main contribution of our work is to show that the classical results for complete logic programs with standard unification can be extended to refer to complete programs equipped with a *stronger* equality theory, still modeling unification. Such a theory, denoted U^*_{DCA}, is obtained by adding to Clark's equality theory U^* an extra axiom, referred to in the literature as the *domain-closure axiom* (DCA) [7, 8]. Informally, the effect of this new axiom is to enforce any model of $(P^*, U^*_{\mathrm{DCA}})$ to have an interpretation domain in which each object must be constructed using the (interpretations of) constant and function symbols. Thus U^*_{DCA}-interpretations are more closely linked to Herbrand interpretations than usual U^*-interpretations.[1] In particular, we will show that $(P^*, U^*_{\mathrm{DCA}})$ and (P^*, U^*) share the same ground (positive and negative) logical consequences, that is, for each ground atom A, $(P^*, U^*_{\mathrm{DCA}}) \models A$ iff $(P^*, U^*) \models A$ as well as $(P^*, U^*_{\mathrm{DCA}}) \models \neg A$ iff $(P^*, U^*) \models \neg A$. Moreover, the equivalence with respect to negative consequences is pervasive up to universally quantified negative theorems, in the sense that given a (possibly open) atom $p(\underline{t})$, $(P^*, U^*_{\mathrm{DCA}}) \models \forall \neg p(\underline{t})$ iff $(P^*, U^*) \models \forall \neg p(\underline{t})$. On the other hand, $(P^*, U^*_{\mathrm{DCA}})$ has universally quantified positive consequences that (P^*, U^*) has not, even though they can be precisely characterized in terms of the consequences of (P^*, U^*).

As a consequence, stronger completeness results for both SLD resolution and negation as failure can be stated and proved with reference to $(P^*, U^*_{\mathrm{DCA}})$. Furthermore, we are in the position of claiming that U^*_{DCA} is the strongest theory for syntactic unification (extending Clark's theory) within which completeness can be still obtained. This is due to a recent result by Michael Maher, who showed that U^*_{DCA} is a *complete* theory, in the classical sense that it decides every formula [12].

Even if this result seems interesting per se, it has also a number of implication from an operational viewpoint. The reason for this is that $(P^*, U^*_{\mathrm{DCA}})$ is characterized by an inference rule related to the notion of *term covering*. A covering of a term t is a (possibly infinite) set of terms $\{t_1, \ldots, t_n, \ldots\}$ such that any ground instance of t is also a ground instance of t_i for some i. For example, referring to the constant a and the unary function f, $\{a, f(x)\}$ is a covering of the variable term y. We will show in Section 3 that the following inference rule is correct for $(P^*, U^*_{\mathrm{DCA}})$:

Let $\phi_{[x]}$ be a wff with variable x free, and let $\{t_1, \ldots, t_n\}$ be a *finite* covering of the term t.

If for each i $\forall \phi_{[t_i]}$ is a logical consequence of $(P^*, U^*_{\mathrm{DCA}})$
then $\forall \phi_{[t]}$ is a logical consequence of $(P^*, U^*_{\mathrm{DCA}})$.

[1] Indeed, U^*_{DCA} acts as a first-order (and hence noncategorical) approximation of the so-called closed-domain assumption (see for instance [13]), which actually consists in considering only Herbrand model. DCA coincides with the closed-domain assumption in the case of database-like programs where no function symbol occurs.

This is what we mean by the sentence "the logical consequences of (P^*, U^*_{DCA}) are closed under finite coverings". The above inference rule can be consistently understood as *proof by case analysis*, in the sense that proving a statement for an exhaustive, finite set of subcases (corresponding to a finite covering) is sufficient to prove the statement in full generality. If compared with induction, the above rule is obviously weaker, in the sense that any statement deducible using the proof-by-case-analysis rule can be deduced using induction, but not vice versa.

The mentioned results allow us to exploit this inference rule to prove some novel operational properties of SLD resolution. Firstly, a notion of completeness of SLD resolution with respect to proof by case analysis is obtained, namely, if $\forall p(\underline{t})\theta$ is a theorem of (P^*, U^*_{DCA}), then there exists a finite set of SLD derivations starting from $p(\underline{t})$ which yields a finite covering of $\underline{t}$. Secondly, a sort of *lifting property* for negation as failure is proved, that is, the finite failure of $p(t_i)$ for each element of a finite covering of a term t guarantees the finite failure of $p(t)$ itself. We believe it is interesting that a number of novel operational features of logic programming can be obtained relying on pure model-theoretic arguments.

The paper is organized as follows. Section 1 sets up some preliminary definitions and terminology. In Section 2 the equality theory and the domain-closure axiom are defined, and a characterization of their models is given. Section 3 introduces coverings and the closure-under-coverings property of (P^*, U^*_{DCA}). Section 4 is devoted to the main results about relationships between (P^*, U^*) and (P^*, U^*_{DCA}), and finally, Section 5 exploits these results to devise some novel operational properties.

1. PRELIMINARIES

Here and throughout the paper we will use underlined symbols to denote tuples of objects. For example, $\underline{x}, \underline{x}_i, \ldots$ will stand for tuples of variables, and $\underline{t}, \underline{t}_i, \ldots$ will stand for tuples of terms.

We refer to [3] and [4] for terminology and notation concerning logic programs with equality and generalized unification. There, a logic program is a pair (P, E) where P is a set of definite Horn clauses and E is a definite-clause equality theory. Unification is then defined with respect to theory E and is referred to as *generalized unification*. An *E-unifier* for two terms s and t is a substitution θ such that $E \models s\theta = t\theta$. Using the fact that E is a definite-clause equality theory, it can be shown that E induces over the Herbrand universe HU a finest congruence ($\approx$), and thus HU$/\approx$ can be considered as the intended domain of interpretation. The notions of Herbrand base and Herbrand interpretations can then be easily generalized to those of *E-base* and *E-interpretations*. Similarly, the fixpoint semantics of generalized logic programs is given by defining a continuous map $T_{(P,E)}$ between E-interpretations.

Appropriate generalizations of the notions of derivation sequence, success set, and finite failure set can also be given for generalized logic programs. A (P, E)-*derivation sequence* (as well as a *fair* (P, E)-derivation sequence) is defined in the same way P-derivation sequences are for standard logic programs, where the notion of standard unification has to be replaced with E-unification. As usual, a (P, E)-derivation sequence can be successful, finitely failed, or infinite. Analogously, the success and finite-failure sets for a given logic program (P, E) are defined as

follows:

$SS(P, E) = \{A \mid A$ is a ground atom and there exists a successful (P, E)-derivation sequence of $A\}$,

$FF(P, E) = \{A \mid A$ is a ground atom and for any fair selection rule there exists a number n such that all (P, E)-derivation sequences of A are finitely failed with length $\leq n\}$,

$GF(P, E) = \{A \mid A$ is a ground atom and for any fair selection rule all (P, E)-derivation sequences of A are finitely failed (their length is unbounded)$\}$,

$GGF(P, E) = \{A \mid A$ is a ground atom and for any fair selection rule all ground (P, E)-derivation sequences of A are finitely failed$\}$.

Note that $A \in GGF(P, E) \setminus FF(P, E)$ means that A has an infinite (fair) derivation sequence but all its ground derivations are finitely failed. We will write simply $SS(P), FF(P), \ldots$ in the case of standard logic programs.

When negation as failure is used to obtain negative information from a standard logic program, it is well known [1, 9] that *complete* logic programs are needed in order to give negation as failure a declarative meaning. Indeed, the use of the very same rule in the framework of generalized logic programs with equality requires a suitable generalization of the notion of complete programs. This is achieved in [3] by defining the completion (P^*, E^*) of a logic program (P, E), where P^* is an *augmented definite-clause program* obtained from P using essentially Clark's transformation, whereas E^* is a *unification complete* equality theory[2] extending E. As an instance, Clark's theory U^* for equality is a unification complete extension of the equality theory consisting only of the usual equality axioms. The following proposition states the soundness and completeness of successful (failed) (P, E)-derivations for positive (negative) ground atoms valid in (P^*, E^*).

Proposition 1.

(i) $(P^*, E^*) \models A$ *iff* $A \in SS(P, E)$.

(ii) $(P^*, E^*) \models \neg A$ *iff* $A \in GF(P, E)$.

As one would expect, these results mirror the corresponding ones for standard logic programs, although the finite failure set FF should be replaced here by the general finite failure set GF. Indeed, when the theory E is such that for all pairs of terms s and t there exists a *finite* set of maximally general E-unifiers, then $FF(P, E) = GF(P, E)$. This is obviously true when E-unification is just standard unification, which will be always the case throughout this paper.

2. EQUALITY THEORY AND DOMAIN-CLOSURE AXIOM

From now on, we will refer to logic programs defined over a first-order language providing a set Σ of function symbols (constants are just 0-adic functions) and a set Π of nonlogical predicate symbols not containing "$=$". Let U^* be the following

[2] An equality theory E^* is *unification complete* if for every equation $s = t$, $E^* \models ((s = t) \rightarrow \bigvee_i \theta_i)$, where $\{\theta_i\}$ is the (possibly empty or infinite) set of E^*-unifiers of s and t, looked upon as a set of equations. Whenever this set is empty, the disjunction is assumed to be false.

unification complete equality theory:

(= 1) $\forall x(x = x)$.

(= 2) $\forall xyzw((x = y \wedge z = w) \rightarrow (x = z \rightarrow y = w))$.

(= 3) $\forall x_1, \ldots, x_n, y_1, \ldots, y_n (x_1 = y_1, \ldots, x_n = y_n \leftrightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n))$ for each n-ary function $f \in \Sigma$, $n \geq 0$.

(= 4) $\forall x_1, \ldots, x_n, y_1, \ldots, y_m (f(x_1, \ldots, x_n) \neq g(y_1, \ldots, y_m))$ for each pair of functions $f, g \in \Sigma$, $f \neq g$.

(= 5) $\forall x(t \neq x)$ for each nonvariable term t such that x occurs in t.

Axioms (= 1),...,(= 5) are actually the equality theory underlying *unification*, introduced in [1]. (= 1) is the axiom for reflexivity of =, while (= 2) is the axiom for both symmetry and transitivity. (= 3) ensures that each function symbol must be interpreted as a function (if part) and also that this function must be injective (only if part). (= 4) ensures that objects constructed using different leftmost constructors are different, that is, each function (respectively, constant) symbol has to be interpreted as a distinct function (object). (= 5) states that if t is a subterm of t', then t and t' must be mapped into different objects: This corresponds to the so-called *occur check* in the unification algorithm.

Let now U^*_{DCA} be U^* extended with the following axiom:

$$\text{(DCA)}\quad \forall x \Big(\bigvee_{f \text{ is a function}} \exists y_1, \ldots, y_n (x = f(y_1, \ldots, y_n)) \vee \bigvee_{c \text{ is a constant}} x = c \Big).$$

This last axiom is referred to in [7, 8] as the *domain-closure axiom*. It forces the interpretation domain of a model to contain only objects which can be constructed (even if not effectively) using the (interpretation of) constant and function symbols.

REMARK. U^*_{DCA}-unification is nothing but standard unification, and thus U^*_{DCA} is unification complete.

It was argued in the introduction that U^*_{DCA}-interpretations are more closely linked to Herbrand interpretations than to U^*-interpretations. This is motivated by the following theorem, where we use $|M|$ for the domain of a model M, HU_Σ for the Herbrand universe over Σ and $[f]_M$ for the interpretation in M of the symbol f.

Theorem 1. Let $M \models U^*_{\mathrm{DCA}}$. *Then*

(a) $|M|$ *contains an isomorphic copy* H *of the Herbrand universe* HU_Σ.

(b) $d \in |M| \setminus H$ *iff* $d = [f]_M(d_1, \ldots, d_m)$ *for some function* f *in* Σ, $m \geq 1$, *and at least one* $d_j \in |M| \setminus H$.

PROOF. (a): H is inductively constructed as the least subset of $|M|$ which satisfies:

(i) $[c]_M \in H$ for each constant symbol c of Σ;

(ii) $[f]_M(d_1, \ldots, d_m) \in H$ for each function symbol f of Σ if $d_1, \ldots, d_m \in H$.

H is clearly isomorphic to HU_Σ by axioms (= 3), (= 4), and (= 5).

(b): Given $d \in |M| \setminus H$, first of all notice that for each constant symbol c of Σ, $d \neq [c]_M$. Hence, $d = [f]_M(d_1, \ldots, d_m)$ for some function symbol f of Σ, by (DCA).

Suppose that $\{d_1, \ldots, d_m\} \subset H$: then, by construction, $d \in H$, contradicting the hypothesis $d \in |M| \setminus H$. Conversely, U^* ensures that each object in H is constructed in a unique way from (interpretation of) constants and functions. Hence, if $d = [f]_M(d_1, \ldots, d_m)$ with $d_j \in |M| \setminus H$, d cannot belong to H. □

In other words, by Theorem 1 we have that if $M \models U^*_{\mathrm{DCA}}$ then for each $d \in |M|$ there exists a nonvariable term t over Σ and a variable assignment V such that $d = [t]_{M,V}$. Notice that (a) holds also for U^*.

As an example, let Σ_{Nat} contain the constant 0 and the unary function s; thus, the corresponding equality theory U^*_{DCA} is

$(=1)$ $\forall x(x = x)$.

$(=2)$ $\forall xyzw((x = y \wedge z = w) \rightarrow (x = z \rightarrow y = w))$.

$(=3)$ $\forall xz(x = z \leftrightarrow s(x) = s(z))$.

$(=4)$ $\forall x(s(x) \neq 0)$.

$(=5)$ $\forall x(s^n(x) \neq x)$, $n \geq 1$.

(DCA) $\forall x(x = 0 \vee \exists z.x = s(z))$.

The Herbrand universe, consisting of ω (the set of natural numbers) with the obvious interpretation for 0 and s, is of course a model of both U^* and U^*_{DCA}. On the other hand, $\omega \cup \{s^n(\cdot)\}_{n \in \omega}$, with $\cdot$ distinct from any natural number, is a model of U^* but not of U^*_{DCA}, since $\cdot$ is not constructed via s from any other object. Consider instead the interpretation M consisting of the disjoint union $\omega + Z$, where Z is the set of (positive and negative) integers (disjoint union avoids confusion between naturals and nonnegative integers). Moreover, define an appropriate successor function Succ on $\omega + Z$, which yields either the natural or integer successor depending on the argument. By defining $[0]_M = 0 \in N$ and $[s]_M =$ Succ, M is a model of U^*_{DCA}. Notice that in this case it is impossible to effectively denote an object in Z by means of a ground term of the language, even if an open term like $s(x)$ has objects in Z as M-instances. Finally, notice that this model satisfies DCA, since also for each $m \in Z$ there exists an object z ($\in Z$) such that $m = [s]_M(z)$.

3. COVERINGS

This section introduces *term coverings*, the crucial notion used in the sequel to relate the various results of this paper to each other.

Following [6], let ground(t) denote the set of ground instances of a term t. If t' is an instance of t (written $t' \leq t$), i.e. there exists a substitution θ such that $t\theta = t'$, then ground(t') $\subseteq$ ground(t). We say that t and t' are equivalent (written $t \sim t'$) if $t' \leq t$ and $t \leq t'$. A substitution $\gamma = \{x_1 \rightarrow y_1, \ldots, x_n \rightarrow y_n\}$ is a *permutation of variables* if the y_i's are distinct variables and $\{x_1, \ldots, x_n\} = \{y_1, \ldots, y_n\}$. The term t is a *variant* of t' if there exists a permutation of variables γ such that $t\gamma = t'$. If the Herbrand universe is not trivial (i.e. it is neither empty nor a singleton), then $t \sim t'$ iff ground(t) = ground(t') iff t is a variant of t'. Finally, t' is a *proper instance* of t (written $t' < t$) if $t' \leq t$ but t' is not a variant of t. Clearly, $t' < t$ implies ground(t') $\subset$ ground(t).

Let t be a term, and ξ a (possibly infinite) set of terms. Then ξ is a *covering of* t iff for each Herbrand ground instance $g \leq t$ there exists $t' \in \xi$ such that $g \leq t'$. Moreover, ξ is an *exact covering* of t iff ξ is a covering of t and for each $t' \in \xi$, $t' \leq t$. Hence if ξ is a covering of t, then $\text{ground}(t) \subseteq \bigcup_{t' \in \xi} \text{ground}(t')$, and if ξ is an exact covering of t, then $\bigcup_{t' \in \xi} \text{ground}(t') = \text{ground}(t)$. A covering for the variable term x is called a *domain covering*. It should be stressed that:

(1) if ξ is an (exact) covering of t, then ξ is an (exact) covering of any variant u of t;

(2) if ξ is an (exact) covering of t, then ξ', obtained from ξ by replacing some term $t' \in \xi$ with a variant u' of t', is an (exact) covering of t.

The notion of (proper) instance and (exact) covering can be naturally extended to n-tuples $(t_1, \ldots, t_n)$ of terms.

REMARK. The notion of term covering, as well as the results relying on it, makes sense only if the Herbrand universe is nonempty, as they refer to the existence of ground terms. Thus, in what follows, it is assumed that the Herbrand universe is nonempty.

A few examples are the following. Referring to Σ_{Nat} introduced in Section 2, $\{x\}, \{0, s(x)\}, \{0, s(0), s(s(x))\}$ are all domain coverings. The set $\{s(0), s(s(x))\}$ is an exact covering of the term $s(x)$, and it is a (nonexact) covering of any term $t \leq s(s(x))$. The set $\{(0, s(0)), (s(x), s(s(x)))\}$ is a covering of the pair of terms $(x, s(x))$. Referring to a signature Σ providing the constant a and the binary function f, the set $\{f(a, a), f(f(x, y), f(z, w))\}$ is a covering of the term $f(x, x)$, and of course it is not a domain covering. The set $\{(f(a, a), x), (f(f(x, y), f(v, w)), z)\}$ is a covering of the pair of terms $(f(x, x), x)$.

The following proposition points out a relation between coverings and models of U^*_{DCA}, namely that *finite* domain coverings actually *cover* the domain of any model of U^*_{DCA}.

Proposition 2. Given Σ and the corresponding equality theory U^*_{DCA}, let $M \models U^*_{DCA}$, and ξ be a finite domain covering. Then for each $d \in |M|$ there exists a term $t \in \xi$ and a variable assignment V such that $d = [t]_{M,V}$.

PROOF. The proof is immediate if the Herbrand universe HU_Σ is finite (i.e. Σ provides only constant symbols), since in this case the interpretation domain of each model of U^*_{DCA} is forced to be isomorphic to HU_Σ.

In the remainder of the proof, the notions of *depth* of a subterm within a term and of *length* of a term are used. The former is defined as the greatest level of occurrence of a subterm within a term, with respect to the ordinary parse tree of the term. The length of a term is defined as follows:

$$\text{length}(t) = 1 \qquad \text{if } t \text{ is a variable or a constant,}$$

$$\text{length}(f(t_1, \ldots, t_n)) = 1 + \max\{\text{length}(t_i)\} \qquad \text{if } f \text{ is a function.}$$

If HU_Σ is infinite, take an arbitrary $M \models U^*_{\text{DCA}}$, such that $|M|$ is not isomorphic to HU_Σ. Take an arbitrary $d \in |M|$. by Theorem 1(a) $|M|$ contains an isomorphic copy H of HU_Σ: If $d \in H$, the proof is immediate. Otherwise, if $d \in |M| \setminus H$, by

Theorem 1(b) $d = [f]_M(d_1, \ldots, d_m)$ for some function symbol f and at least one $d_j \in |M| \setminus H$. Thus, it is possible to construct an open term u_1 of the kind $f(t_1, \ldots, t_m)$ where t_i is a new variable x_i if $d_i \in |M| \setminus H$, and t_i is the ground Herbrand term corresponding to d_i if $d_i \in H$. Clearly, $d = [u_1]_{M,V_1}$ for some variable assignment V_1. Hence, repeated applications of Theorem 1(b) allow one to construct a term u_n, with $d = [u_n]_{M,V_n}$ for some assignment V_n, such that the depth of each variable in u_n is greater than the length of each term in ξ. Replace the variables in u_n by constants to obtain a ground term s. Some term in ξ must cover s, and so it must also cover u_n. □

It is worth noting that the above proposition no longer holds if one drops the condition that ξ is finite. As an example, consider the (infinite) domain covering $\{0, s(0), s(s(0)), \ldots\}$ with respect to Σ_{Nat}: clearly, it does not cover any domain of a model of U^*_{DCA} which is not isomorphic to $\text{HU}_{\Sigma_{\text{Nat}}}$.

Proposition 2 justifies the introduction of an inference rule for theories equipped with U^*_{DCA}, which can be consistently understood as proof by case analysis. In fact Proposition 2 ensures that, exploiting the domain-closure axiom, a finite covering of a term t is representative for all the possible instances of t over any interpretation domain. Hence proving a statement for an exhaustive, finite set of subcases (corresponding to a finite covering) is sufficient to prove the statement in full generality. The following theorem formalizes the above rule in the case of complete logic programs under U^*_{DCA}.

Theorem 2. Let P be a definite program, and $\phi_{[x_1,\ldots,x_n]}$ be a formula, over the same language as P, whose free variables are $x_1, \ldots, x_n$. Let ξ be a finite covering of the n-tuple of terms $(t_1, \ldots, t_n)$. Then

$(P^*, U^*_{\text{DCA}}) \models \forall \phi_{[t'_1,\ldots,t'_n]}$ for each n-tuple $(t'_1, \ldots, t'_n) \in \xi$ if $(P^*, U^*_{\text{DCA}}) \models \forall \phi_{[t_1,\ldots,t_n]}$.

PROOF. Trivial, since from Proposition 2 (extended in a straightforward way to n-tuples of terms) one obtains that, given an arbitrary model M of (P^*, U^*_{DCA}), any M-instance of $(t_1, \ldots, t_n)$ can be obtained as a M-instance of some n-tuple $(t'_1, \ldots, t'_n) \in \xi$. □

As an example, if P is

$p(s(x))$:-

$p(0)$:-

then, since $p(0)$ and $\forall x p(s(x))$ are both logical consequences of (P^*, U^*_{DCA}) and $\{0, s(x)\}$ is a covering of the term x, $\forall x p(x)$ is also a logical consequence of (P^*, U^*_{DCA}). It is worth noting that $\forall x p(x)$ is not a logical consequence of (P^*, U^*).

In the rest of the paper we will exploit these properties in order to characterize the relationships between (P^*, U^*_{DCA}) and (P^*, U^*), as well as to derive some useful properties of SLD resolution.

4. COMPLETE LOGIC PROGRAMS AND U^*_{DCA}

In this section, the results summarized in Proposition 1 are rephrased when (P^*, U^*) is replaced with (P^*, U^*_{DCA}). Furthermore, exploiting the proof techniques used in [3] and the notions introduced in the previous section, stronger completeness results will be provided for both SLD resolution and negation as failure.

First of all, by observing that U^*_{DCA}-unification is nothing but standard unification and that $GF(P, E) = FF(P, E)$ as far as E-unification is standard unification, Proposition 1 has the following

Corollary 1.

(i) $(P^*, U^*_{DCA}) \models A$ *iff* $A \in SS(P)$.

(ii) $(P^*, U^*_{DCA}) \models \neg A$ *iff* $A \in FF(P)$.

Corollary 1 ensures that (P^*, U^*) and (P^*, U^*_{DCA}) share the same (positive and negative) ground logical consequences. Indeed, this equivalence is stronger as far as negative consequences are taken into account, as we show below.

Theorem 3. Let P be a program and $p(\underline{t})$ be an atom. The following statements are equivalent:

(i) $(P^*, U^*_{DCA}) \models \forall \neg p(\underline{t})$;

(ii) $(P^*, U^*) \models \forall \neg p(\underline{t})$;

(iii) $p(\underline{t})$ *has only finitely failed fair SLD derivations from P.*

PROOF. The equivalence of (ii) and (iii) summarizes soundness [1] and completeness [2, 14] of negation as failure.

The equivalence between (i) and (iii) can easily be shown as follows. Let Ans be a propositional symbol not occurring in P, and $P+$ the program obtained by adding the clause $Ans :- p(\underline{t})$ to P. Then $p(\underline{t})$ is finitely failed in P iff $Ans \in FF(P+)$ iff [by Corollary 1(ii)] $(P+^*, U^*_{DCA}) \models \neg Ans$ iff $(P+^*, U^*_{DCA}) \models \forall \neg p(\underline{t})$. □

The above result is actually a stronger form of completeness for negation as failure, exploiting the equivalence between (P^*, U^*) and (P^*, U^*_{DCA}) with respect to the negative information. Nevertheless, it should be stressed that such an equivalence holds for formulae of the kind $\forall \neg p(\underline{t})$ only. For instance, let P be

$p(s(x)) :-$

$p(0) :-$

Then, $\forall x p(x)$ is a logical consequence of (P^*, U^*_{DCA}), but it is not a logical consequence of (P^*, U^*). Nevertheless, the notion of (finite) coverings allows us to characterize precisely which formulae of the kind $\forall p(\underline{t})$ are logical consequences of (P^*, U^*_{DCA}) but are not logical consequences of (P^*, U^*). This is achieved by the next theorem.

Theorem 4. Let P be a logic program and $p(\underline{t})$ be a (possibly open) atom. Then $(P^*, U^*_{DCA}) \models \forall p(\underline{t})$ iff there exists a finite covering $\{\underline{t}'_1, \ldots, \underline{t}'_k\}$ of $\underline{t}$ such that $(P^*, U^*) \models \forall p(\underline{t}'_i)$ for each $i = 1, \ldots, k$.

PROOF. $\leftarrow$: Straightforward, observing that $(P^*, U^*_{DCA}) \models \forall p(\underline{t}'_i)$ for each $i = 1, \ldots, k$, and thus Theorem 2 applies, yielding $(P^*, U^*_{DCA}) \models \forall p(\underline{t})$.

$\rightarrow$: Suppose that for each finite covering $\{\underline{t}'_1, \ldots, \underline{t}'_k\}$ of $\underline{t}$ there exists i in $1, \ldots, k$ such that $(P^*, U^*) \cup \exists \neg p(\underline{t}'_i)$ has a model. This is also true for the finite covering of $\underline{t}$ given by $\{\underline{t}\}$. Moreover, by Corollary 1, for each ground instance A of $p(\underline{t})$, $(P^*, U^*) \models A$. The completeness property of SLD resolution guarantees that, for such an A, $p(\underline{t})$ has a successful derivation with answer substitution σ such that $A \leq p(\underline{t})\sigma$. Let $\{\sigma_i\}$ be the set of all distinct computed substitutions. This set must be infinite, since otherwise the set $\{\underline{t}\sigma_i\}$ would constitute a finite covering of $\underline{t}$. This implies that $p(\underline{t})$ is the first goal of an infinite derivation sequence. Following the proof technique in Theorem 6 of [3] and recalling that U^*_{DCA}-unification is nothing but standard unification, such an infinite derivation sequence allows us to construct an extension of U^*_{DCA}, say $U^*_{DCA}+$, that is consistent. This is done by looking at the infinite collection of substitutions $\{\theta_i\}$ in the derivation sequence as a collection of ground equations over a larger alphabet $\Sigma+$ obtained by replacing each distinct variable x_j with a new constant symbol c_j. That is, if the substitution θ_i contains the equations $\{x_1 = s_1(\underline{x}), \ldots, x_k = s_k(\underline{x})\}$, the set E_i of ground equations $\{c_1 = s_1(\underline{c}), \ldots, c_k = s_k(\underline{c})\}$ is constructed, where $c_1, \ldots, c_k$ are new constant symbols. Then $U^*_{DCA}+$ is obtained as $U^*_{DCA} \cup \{E_i\}$. It can be shown as in [3] that $U^*_{DCA} \cup \{E_i, \ldots, E_n\}$ is a conservative extension of U^*_{DCA} and thus it is consistent. Then, the compactness theorem allows us to state that also $U^*_{DCA}+$ is consistent. Finally, the infinite derivation sequence can be looked upon as an infinite ground $(P^*, U^*_{DCA}+)$-derivation sequence of a ground instance of $p(\underline{t})$. Thus, such a ground instance is true in some $U^*_{DCA}+$ models and false in others, since it belongs to $GGF(P, U^*_{DCA}+) \setminus FF(P, U^*_{DCA}+)$ [4]. This contradicts the hypothesis that $(P^*, U^*_{DCA}) \models \forall p(\underline{t})$, since $U^*_{DCA}+$ models are of course U^*_{DCA} models. □

Intuitively, the previous result points out that the only logical consequences of complete programs of the form $\forall p(\underline{t})$ induced by the domain closure axioms are those provable by case analysis (see Section 3). As a consequence a completeness result for SLD resolution with respect to (P^*, U^*_{DCA}) can be stated.

Corollary 2. Let P be a logic program, $\underline{t} \leq \underline{t}'$ terms, and p a predicate symbol such that $(P^*, U^*_{DCA}) \models \forall p(\underline{t})$. Then $p(\underline{t}')$ has a finite set of successful SLD derivations with answer substitutions $\{\sigma_1, \ldots, \sigma_k\}$ such that $\{\underline{t}'\sigma_1, \ldots, \underline{t}'\sigma_k\}$ is a finite covering of $\underline{t}$.

PROOF. Straightforward from Theorem 4 and completeness of SLD resolution with respect to (P^*, U^*). □

The above corollary can also be obtained as a consequence of Theorem 2 of [11], considering U^*_{DCA} as the theory D axiomatizing the domain and the constraints as the equations corresponding to computed substitutions.

It is worth noting that SLD resolution is complete with respect to (P^*, U^*_{DCA}) in a weaker sense than it is with respect to (P^*, U^*). Nevertheless, Theorems 3 and 4 together enable us to claim that *complete logic programs with DCA* are a stronger (and hence more informative) theoretical framework for logic programming with negation as failure than standard complete logic programs are. An outcome of our results is a deeper insight into some operational issues, which is the matter of the next section.

5. LIFTING PROPERTIES

From the operational viewpoint, the main consequence of the results of the previous sections is a property for negation as failure, which is actually the dual property of the lifting lemma for successful SLD refutations. In fact, exploiting the proof-by-case-analysis rule and the notion of covering, the finite failure over a finite, exhaustive set of subcases is sufficient to guarantee the finite failure in the general case.

Given a program P, we will write

"$p(\underline{t})$ *fails*" as an abbreviation for "$p(\underline{t})$ has only finitely failed (fair) derivations";

"$p(\underline{t})$ *succeeds*" as an abbreviation for "$p(\underline{t})$ has a successful (fair) derivation";

"$p(\underline{t})$ *diverges*" as an abbreviation for "$p(\underline{t})$ has no successful (fair) derivations and has at least one infinite derivation".

The following statement formalizes the lifting property mentioned above.

Theorem 5. Let P be a program and $p(\underline{t})$ be an atom. Let ξ be a finite covering of $\underline{t}$. If $p(\underline{t}')$ fails for each $\underline{t}' \in \xi$, then $p(\underline{t})$ fails.

PROOF. $p(\underline{t}')$ fails for each $\underline{t}' \in \xi$ implies (by Theorem 3)

$(P^*, U^*_{\mathrm{DCA}}) \models \forall \neg p(\underline{t}')$ for each $\underline{t}' \in \xi$, which implies (by Theorem 2)

$(P^*, U^*_{\mathrm{DCA}}) \models \forall \neg p(\underline{t})$,

which implies (by Theorem 3)

$p(\underline{t})$ fails. □

Notice that the above property is false if one relaxes the condition on finiteness of the covering. For instance, let P (over Σ_{Nat}) be defined by the single clause

$$p(s(x)) \text{:-} p(x)$$

Then $p(s^n(0))$ fails for each $n \in \omega$ and $\{s^n(0))\}_{n \in \omega}$ is a covering of the variable term x, but $p(x)$ diverges.

The following is a particular case of the above theorem when no function symbols are used in the program.

Corollary 3. Let P be a program over Σ providing no function symbol, and $p(\underline{t})$ be an atom. Then $p(\underline{s})$ fails for each $\underline{s} \in \mathrm{ground}(\underline{t})$ only if $p(\underline{t})$ fails.

PROOF. It is a consequence of Theorem 5, since ground($\underline{t}$) is a finite covering of t. □

The following is a further, weaker lifting property of finite failure, which will be useful shortly.

Corollary 4. Let P be a program and $p(\underline{t})$ be a nonground atom. For each $\underline{t}' < \underline{t}$, $p(\underline{t}')$ fails only if $p(\underline{t})$ fails.

PROOF. Pick up from the set of proper instances of $\underline{t}$ a finite (exact) covering of $\underline{t}$, and apply Theorem 5. □

The next two results exploit the lifting property of the finite failure to devise a converse property of divergence.

Corollary 5. Let P be a program and $p(\underline{t})$ be a nonground atom. If $p(\underline{t})$ diverges, then there exists a proper instance $\underline{t}'$ of $\underline{t}$ such that $p(\underline{t}')$ diverges.

PROOF. Suppose that for each proper instance $\underline{t}'$ of $\underline{t}$ $p(\underline{t}')$ does not diverge. The following two cases must be considered:

(a) For some proper instance $\underline{t}'$ of $\underline{t}$, $p(\underline{t}')$ succeeds. This gives a contradiction, since in this case $p(\underline{t})$ succeeds too.

(b) For each proper instance $\underline{t}'$ of $\underline{t}$, $p(\underline{t}')$ fails. Then apply Corollary 4 to get a contradiction. □

Corollary 6. Let P be a definite program over Σ providing no function symbol, and $p(\underline{t})$ be an atom. If $p(\underline{t})$ diverges, then there exists a ground instance $\underline{s}$ of $\underline{t}$ such that $p(\underline{s})$ diverges.

PROOF. Suppose that for each ground instance $\underline{s}$ of $\underline{t}$, $p(\underline{s})$ does not diverge. The following two cases must be considered:

(a) For some ground instance $\underline{s}$ of $\underline{t}$, $p(\underline{s})$ succeeds. This gives a contradiction, since in this case $p(\underline{t})$ succeeds too.

(b) For each ground instance $\underline{s}$ of $\underline{t}$, $p(\underline{s})$ fails. Then apply Corollary 3 to get a contradiction. □

REMARK (due to J.-L. Lassez). Corollaries 3 and 6 have been obtained using the notion of term covering. Following a different approach, they can be generalized by taking into account the notion of *canonical* programs, that is, programs such that FF(P) = GGF(P) [5]. More precisely, Corollary 3 can be generalized as follows:

Let P be a program and $p(\underline{t})$ an atom such that $P \cup \{\text{Ans} :\text{-} p(\underline{t})\}$ is canonical (where Ans *is a new propositional constant). Then $p(\underline{s})$ fails for each $\underline{s} \in \text{ground}(\underline{t})$ only if $p(\underline{t})$ fails.*

The result easily follows by noting that every infinite fair derivation for $p(\underline{t})$ would be also an infinite derivation for Ans. Similar considerations apply to Corollary 6.

6. CONCLUSIONS

Complete logic programs augmented with the domain-closure axiom have been proposed, and related completeness results for logic programming with negation as failure have been proved. In [12] it is proved that U^*_{DCA} is a complete theory, provided that the alphabet contains at least two function symbols. Therefore, all these results imply that complete logic programs with DCA are not only a stronger

framework than ordinary complete logic programs, but indeed the strongest framework (embedding Clark's equality axioms) within which completeness is preserved. This should constitute a good motivation to take complete logic programs with DCA as *the* reference theory for logic programming with negation as failure. Moreover, the proof-by-case-analysis inference rule, induced by DCA, gave an opportunity to reach a deeper insight into the operational properties of SLD resolution.

Herbrand models obviously satisfy DCA, but unfortunately it is not possible to achieve completeness for negation as failure by restricting attention to them. DCA brings one closer to Herbrand models than Clark's equality theory alone does, still preserving completeness. Open issues are related to this latter remark: It would be nice to devise some notion of *canonical* interpretation domain for U^*_{DCA}, say D, such that models over D are enough to preserve completeness (of course, such a D has to properly include the Herbrand universe). Perhaps interesting links with this issue can be found in the area of recursive domain equation solving.

Some proofs in the early version of this paper were long, tedious and, to a large extent, unnecessary. We are greatly indebted to Jean-Louis Lassez and Michael Maher, who gave us a lot of suggestions for simplifying the proofs and turning a collection of theorems into a paper.

REFERENCES

1. Clark, K. L. Negation as Failure, in: H. Gallaire and J. Minker, (eds.), *Logic and Data Bases*, Plenum, New York, 1978, pp. 293–322.
2. Jaffar, J., Lassez, J.-L., and Lloyd, J. W., Completeness of the Negation-as-Failure Rule, in: *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, 1983, pp. 500–506.
3. Jaffar, J., Lassez, J.-L., and Maher, M. J., A Theory of Complete Logic Programs with Equality, *J. Logic Programming* 3:211–223 (1984).
4. Jaffar, J., Lassez, J.-L., and Maher, M. J., A Logic Programming Language Scheme, in: D. De Groot and G. Lindstrom (eds.), *Logic Programming: Relations, Functions and Equations*, Prentice-Hall, 1985.
5. Jaffar, J. and Stuckey, P. J. Canonical Logic Programs, *J. Logic Programming* 2:143–155 (1986).
6. Lassez, J.-L., Maher, M. J., and Marriot, K., Unification Revisited, Draft, IBM T. J. Watson Research Center, 1986.
7. Lloyd, J. W. and Topor, R. W., A Basis for Deductive Data Base Systems, *J. Logic Programming* 2:93–103 (1985).
8. Lloyd, J. W. and Topor, R. W. A Basis for Deductive Data Base Systems. II, *J. Logic Programming* 3:55–67 (1986).
9. Lloyd, J. W., *Foundations of Logic Programming*, Springer Symbolic Computation Series, Berlin, 1984.
10. Lloyd, J. W., *Foundations of Logic Programming*, 2nd ed., draft, 1987.
11. Maher, M. J., Logic Semantics for a Class of Committed-Choice Programs, in: *Proceedings of the Fourth International Conference on Logic Programming*, Melbourne, 1987, pp. 858–876.

12. Maher, M. J., Complete Axiomatizations of the Algebras of Finite, Infinite and Rational Trees, Technical Report, draft, IBM T. J. Watson Research Center, Yorktown Heights, N.Y., 1987.
13. Shepherdson, J. C., Negation as Failure: A Comparison of Clark's Completed Data Base and Reiter's Closed World Assumption, *J. Logic Programming* 1:51–79 (1984).
14. Wolfram, D. A., Maher, M. J., and Lassez, J.-L., A Unified Treatment of Resolution Strategies for Logic Programs, in: *Proceedings of the Second International Conference on Logic Programming*, Uppsala, 1984, pp. 263–276.